Tarea 3 – Generador y Validador de Password

1. Utilice ciclos for, estructuras de control y la librería numpy solamente para crear un generador de password aleatorio con las siguientes características:

Básese en <https://passwordsgenerator.net/> para mejor entendimiento

* Admite números, letras y alfanuméricos
* El primer carácter es una letra o un alfanumérico
* Solamente permite 3 números
* El largo mínimo es 8 y el máximo es 15
* Números, letras y alfanuméricos no se repiten
* Solo admite 3 mayúsculas, mínimo 1 mayúscula

2. Genere una función que valide el password de arriba:

* La salida debe ser True si el password cumple con las validaciones
* Si la salida es False, señalar cuales condiciones no se cumplen para reparar.

3. Genere dos archivos en python que hagan el llamado a la sección 1 y el otro a la sección 2.

Opcional: Sin Nota… Finalmente encripte el password con el valor semilla ‘ulatina’ utilizando la librería cryptography, módulo fernet, función fernet

from cryptography.fernet import Fernet

import base64

import hashlib

def encrypt\_password(seed, password):

key = hashlib.sha256(seed.encode()).digest()

cipher\_suite = Fernet(base64.urlsafe\_b64encode(key))

encrypted\_password = cipher\_suite.encrypt(password.encode())

return encrypted\_password.decode()

def decrypt\_password(seed, encrypted\_password):

key = hashlib.sha256(seed.encode()).digest()

cipher\_suite = Fernet(base64.urlsafe\_b64encode(key))

decrypted\_password = cipher\_suite.decrypt(encrypted\_password.encode())

return decrypted\_password.decode()

# Example usage

seed = "valorsemilla"

password = "SuPassword"

# Encrypt the password

encrypted\_password = encrypt\_password(seed, password)

print("Password Encriptado:", encrypted\_password)

# Decrypt the password

decrypted\_password = decrypt\_password(seed, encrypted\_password)

print("Password desencriptado:", decrypted\_password)